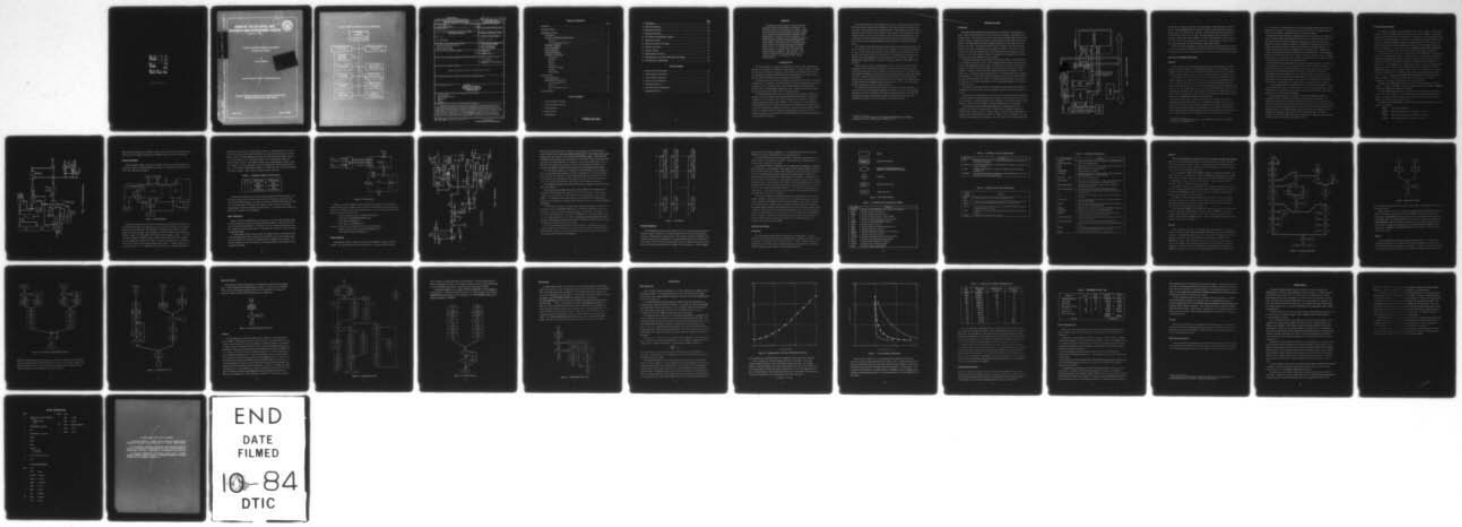
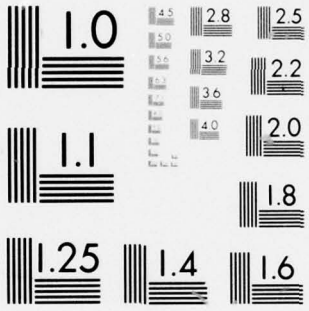
AD-A043 193 A PAGED HARDWARE ASSOCIATIVE MEMORY(U) DAVID W TAYLOR 1/1
NAVAL SHIP RESEARCH AND DEVELOPMENT CENTER BETHESDA MD
J R CARLBERG AUG 77 DTNSRDC-77-0083

UNCLASSIFIED F/G 9/2 NL



END
DATE FILMED
10-84
DTIC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

A PAGED HARDWARE ASSOCIATIVE MEMORY - PRELIMINARY REPORT

Report 77-0083

DAVID W. TAYLOR NAVAL SHIP RESEARCH AND DEVELOPMENT CENTER

Bethesda, Md. 20084

A PAGED HARDWARE ASSOCIATIVE MEMORY
PRELIMINARY REPORT

by

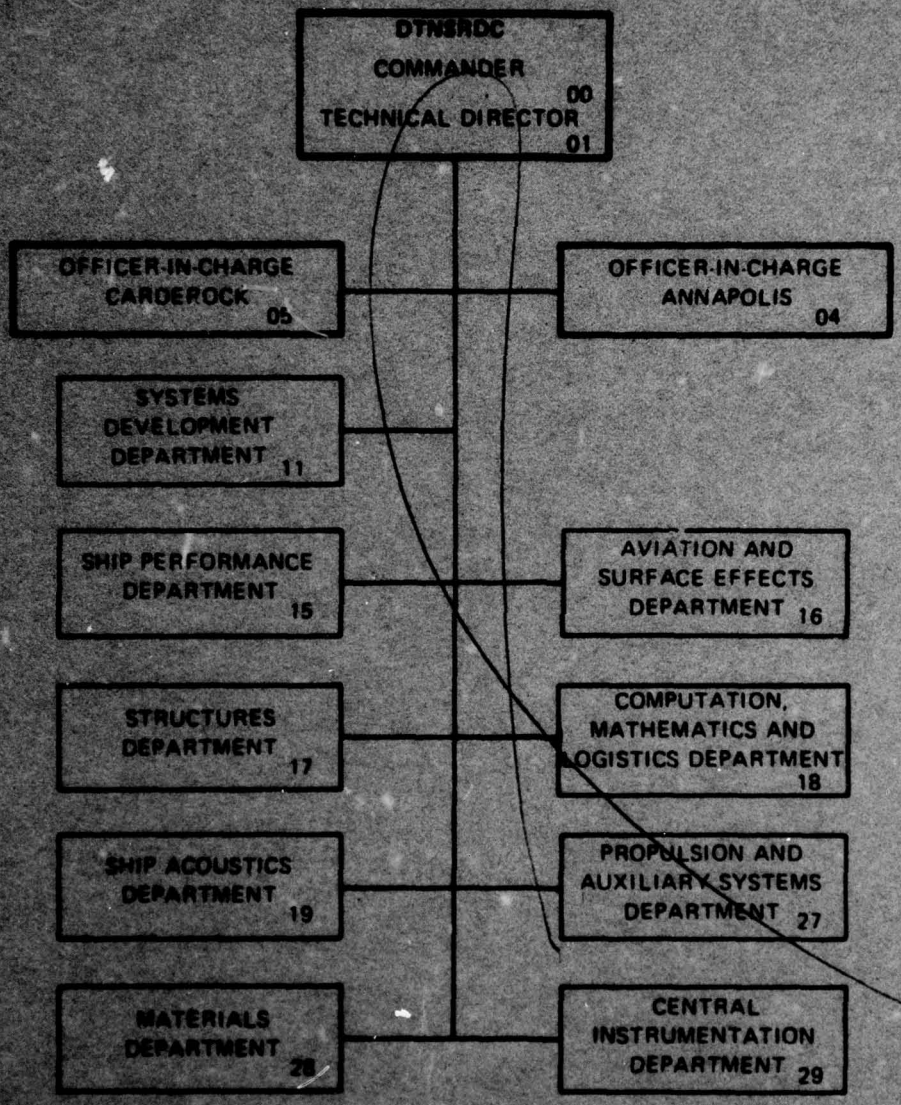
James R. Carlberg

COMPUTATION, MATHEMATICS, AND LOGISTICS DEPARTMENT
RESEARCH AND DEVELOPMENT REPORT

August 1977

Report 77-0083

MAJOR DTNSRDC ORGANIZATIONAL COMPONENTS



UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

AD A043193

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: DTNSRDC Report 77-0083	2. GOVT ACCESSION NO.
3. RECIPIENT'S CATALOG NUMBER	
4. TITLE (and Subtitle): A PAGED HARDWARE ASSOCIATIVE MEMORY. PRELIMINARY REPORT.	5. TYPE OF REPORT & PERIOD COVERED: Preliminary rept.
	6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s): James R. Carlberg	8. CONTRACT OR GRANT NUMBER(s): ZR0134?2
9. PERFORMING ORGANIZATION NAME AND ADDRESS: David W. Taylor Naval Ship Research and Development Center, Bethesda, Maryland 20084	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: ~~Task Area~~ ZR014 0201, Work Unit 1-1834-003
11. CONTROLLING OFFICE NAME AND ADDRESS	12. REPORT DATE: August 1977
	13. NUMBER OF PAGES: 41
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)	15. SECURITY CLASS. (of this report): UNCLASSIFIED
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

APPROVED FOR PUBLIC RELEASE: DISTRIBUTION UNLIMITED

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Associative Memory
Content addressable memories
Paged Memory
Design
Computers

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

A hardware associative memory is a device that can be employed to enhance the performance of computers. The associative memory system described in this report is a paged system which exploits the speed inherent in parallel (random) access capabilities of software associative memories and the storage capacity of lower cost but slower memories. The described system is designed to run as peripheral equipment to a host computer which issues information requests to the memory system. These requests are of a form that permits one to conveniently manipulate information embedded in graph structures. ~~We conclude that~~ a paged associative memory system is an attractive method of implementing associative memory capabilities. This approach is a cost effective, flexible alternative to other approaches such as software simulations or non-paged associative memory implementations.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

A hardware associative memory is a device that can be employed to enhance the performance of computers. The associative memory system described in this report is a paged system which exploits the speed inherent in parallel (random) access capabilities of software associative memories and the storage capacity of lower cost but slower memories. The described system is designed to run as peripheral equipment to a host computer which issues information requests to the memory system. These requests are of a form that permits one to conveniently manipulate information embedded in graph structures. We conclude that a paged associative memory system is an attractive method of implementing associative memory capabilities. This approach is a cost effective, flexible alternative to other approaches such as software simulations or non-paged associative memory implementations.

INTRODUCTION

The work discussed in this report was developed as a result of a Navy requirement for detection and classification of underwater acoustic transients. Innovative hardware configurations and devices to enhance the speed, ease, and efficiency of extracting acoustic features and constructing discrimination algorithms have been investigated and a hardware associative memory capable of capitalizing on this technology has been defined.

A hardware associative memory is an important concept that can be employed to enhance the performance of computers. Associative memories are suited for application areas such as information retrieval, scheduling, traffic control, virtual memories, pattern recognition, machine learning, and large rapidly changing data bases. Associative processors can handle these applications more efficiently than can conventional systems.

An associative memory differs from a standard random access memory (RAM) in that memory cells can be accessed via their contents rather than their addresses. To illustrate, consider the case of an employer who wishes to know which of the group of prospective employees facing him have electronics experience. If the group were to be considered an ordinary RAM, the employer would question each person individually to determine whether or not that person had any experience in electronics. Using the associative memory approach, however, the employer would first address the group collectively, saying, "If you have electronic experience, raise your hand," and would then question only those already identified as having electronic experience.

The associative memory system described in this report is a paged system which exploits the speed in parallel (random) access capabilities of semiconductor associative memories and the storage capacity of lower cost but slower memories. A semiconductor associative memory component costs about $0.31 per bit, whereas a semiconductor charge coupled device (CCD) memory component costs only about $0.0015 per bit. By judiciously mixing the two types of memories a reasonably priced associative memory system can be constructed which will have a useful amount of storage capability and a useful amount of associative processing power.

The design of the associative memory system has taken into consideration the source and type of operating requests. The described system is designed to run as peripheral equipment to a host computer which issues information and data requests to the memory system. The memory system returns results to the host computer. The command set for the associative memory can be easily generated from a GIRL program in the host computer. GIRL[1] is a programming language designed to conveniently manipulate information embedded in graph structures.

Graph structures are used in implementing an acoustic transient classifier. Waveform processing is done by representing a waveform as a string of rises, falls, and levels, and by storing the representations as trees of variation for each transient occurrence. Results from the waveform processing serve as features used to parse decision trees to automatically classify a transient. Currently these structures are processed and stored by a software associative memory simulator which entails memory and execution time overhead. The paged hardware associative memory would perform a graph structure operation in 1/15 of the time required by the existing software associative memory simulator.

The associative memory has other important applications. For example, a memory system can be used in implementing a field hardware classification system. Also, associative memories may be used in conjunction with large computing systems such as the CDC 6700 to efficiently implement virtual memory capability, information retrieval systems, job scheduling, networking, and many other types of nonnumeric processing.

A-35272

[1] Berkowitz, S., *"Graph Information Retrieval Language; Programming Manual for FORTRAN Complement; Revision One,"* DTNSRDC Report 76-0085 (Feb 1976).

PROPOSED SYSTEM

OVERVIEW

The paged hardware associative memory system can be divided into four operational blocks. These operational blocks consist of the control, work memory, paged memory, and response resolver. The work memory consists of the content addressable memory (also called associative memory), page pointer memory, available space and response flags, and the mask registers. The paged memory includes the charge coupled device (CCD) mass memory and the page control logic. The response resolver consists of the logic to select one response or available space cell from among several and to generate its address. The control is a microprocessor that interprets commands from the host computer and then generates the signal sequences to execute the command. When the command has been completed, the microprocessor makes the results available to the host computer.

The particular implementation described in this report (see Figure 1) accesses memory by associating the input data, or "key", in the comparand register with all the words stored in the contents addressable memory (CAM). Associations may be made on the entire key or on specific bit fields of the key, depending upon which bits of the mask are set. Memory responses to inquiries are stored in flip flops called response flags, one flip-flop for each memory cell. A particular response flag will be set if its corresponding memory cell matches the interrogation. A second type of flag, the available space flag, is used to indicate memory cells that contain valid data, and, conversely, to indicate memory cells that can be used to store new data. The available space flag is set if the corresponding memory cell is available for storing new data.

The response resolver monitors both the available space flags and the response flags to determine the location of memory cells that respond to inquiries so that those cells may be accessed.

A mask memory is provided which can store up to four masks for the user's convenience in quickly switching the bit fields that apply for inquiries. This technique allows the user to vary field widths to accommodate different data complexities, and to make an inverse inquiry. The inverse inquiry gives the capability of backing through directed trees and graphs.

The proposed associative memory implementation uses a paged memory system. A system that uses memory composed entirely of CAM would be extremely expensive (i.e., $2.3M for 65K cells where each cell is 48 bits), whereas a system that mixes CAM with less expensive mass memory would be reasonable (i.e., $28K for 65K cells consisting of 256 cells of CAM and the remainder in mass memory). The mass memory is to be implemented using

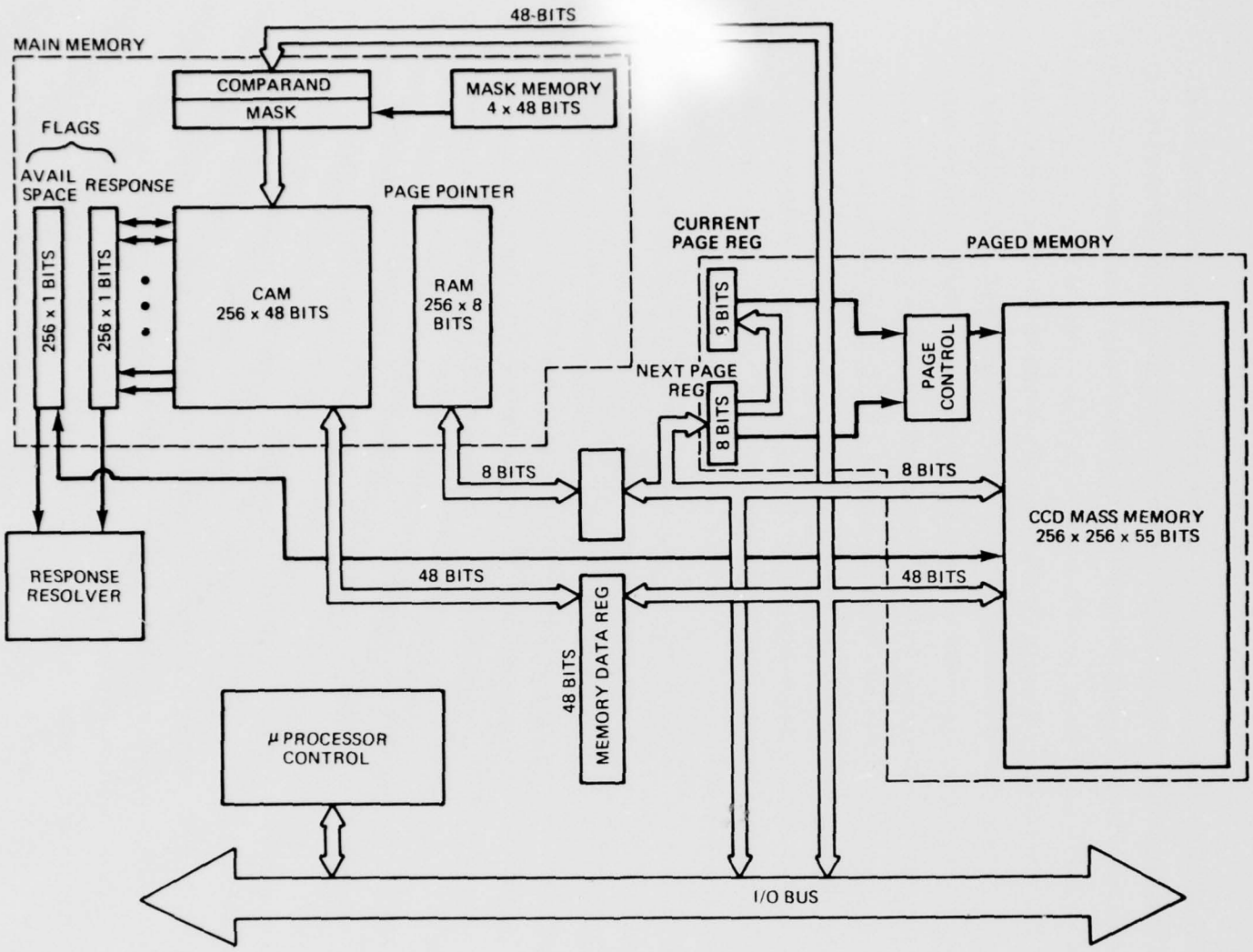


Figure 1 – Associative Memory System

the relatively new charge coupled device (CCD) technology. This technology is inexpensive and fast. Page swapping between the CAM and CCD mass memory should take about 450 microseconds. The system will have 256 pages. An 8-bit page address must be appended to each CAM cell to point to the next page. The page address for each CAM memory cell is stored in a random access memory (RAM). The size of the CCD memory must be 65K cells by 57 bits. The 57 bits are composed of 48 data bits, 8 page-address bits, and one bit for the available space flag. The page control logic handles page swapping.

The entire system is controlled by a microprocessor that receives requests from the I/O bus, executes the request, and returns the result to the I/O bus. Current specifications require a microprocessor with an 8-bit word length and a cycle time of no more than one microsecond.

MULTIPLE RESPONSE RESOLUTION

Overview

When a search of an associative memory yields more than one responder, the multiple-response-resolution problem arises. This problem arises either after an interrogation with multiple responders or when an empty cell must be selected to receive input. The multiple-response-resolution problem is perhaps the least associative process in an associative memory.

The multiple response resolver uses the response store vector as input and generates as output an address vector. The response store vector is a binary array containing two entries for each CAM cell. These entrys are the available space flag and the response flag. The available space flag will be set (to 1) if the corresponding CAM cell is empty; otherwise it will be reset (to 0). The response flag is set to 1 if the corresponding CAM cell matches the last interrogation, and otherwise to 0. The response resolver generates an address pointer to the desired CAM cell. Anderson[2] groups the methods for performing this address generation into three classes—counting schemes, ripple-carry, and logic-trees:

- *Counting* schemes involve a bit-by-bit serial search of the response store vector to determine the location of the desired cell. A counter gives the address of that cell. An advantage of this scheme is that once the first address has been found, the location of the next one can be quickly found just by continuing the count. Moreover the last item can easily be located merely by decrementing the counter and searching in the opposite direction. Also, counting schemes are easy to implement using a reasonable amount of logic.

[2] Anderson, G.A., *"Multiple Match Resolvers: A New Design Method,"* IEEE Trans. on Computers, Vol. C-23, pp. 1317-1320 (Dec 1974).

The disadvantage of counting schemes lies in their slow execution, which results from their dependency on positions of the set bits in the response store vector. Counting schemes lose some of the advantages of the parallel nature of an associative memory.

- Ripple-carry methods duplicate a single logic block at each bit position or each group of positions. Because ripple-carry methods have the same logic for each bit position, they have the advantage of being amenable for large scale integration (LSI) fabrication techniques. Ripple-carry methods, offering the capability of locating the i^{th} response instead of the first response, become quite complex. Look-ahead and look-back capabilities and the use of storage cells (flip-flop) are required to implement an i^{th} response locator. Although more esthetic than counting schemes, these methods are not substantially faster. Long propagation delays occur because logic circuits are connected in series, and a signal may have to propagate through at least as many gates as there are cells in the associative memory. Moreover, ripple-carry methods require more logic than counting schemes.
- Tree-structured logic usually comprises one level of logic blocks associated with elements of the response store vector, with a second level of logic blocks linking the first level, etc., up to three or more levels. Logic trees generally offer the fastest execution time, since gate delays need only propagate through three or four levels. More gates are required for implementing tree-structured logic than for the counting schemes or ripple-carry method. The number of gates required increases combinatorially as the memory cells increase. The logic blocks for tree structured logic are more complex than the logic blocks for the ripple-carry methods. Implementing an i^{th} response location substantially increases complexity.

Two preliminary designs for a multiple response resolver have been completed. The one most favored at present uses the counting scheme. The other, a ripple-carry method, is more expensive than the counting scheme and, since it does not provide significant improvement in performance, is no longer under consideration.

The final design is likely to be a hybrid of two designs, combining the inherent advantages of both the ripple-carry and the tree structured logic implementations. For example, the associative memory could be divided into 64 sections of four cells each. The ripple-carry method could be used to resolve responses in each of the 64 sections, after which the tree structured logic could then be used to resolve the 64-section output generated by the ripple-carry logic. This combined method should execute in about seven delay times, four ascribed to the ripple-carry logic and three to the tree-structured logic. The degree of complexity of the hybrid design would be greater than that of a ripple-carry response resolver and less than that of a tree-structured response resolver.

Counting Response Resolver

An example of a modified counting response resolver is shown in Figure 2. This response resolver locates the I^{th} response where I is an 8-bit number stored in INDEX. This modified response resolver performs either a top-to-bottom or a bottom-to-top search, depending on the state of the search control flip-flop. The mode control flip-flop indicates whether the response resolver is to search the available space flags or the response flags for the I^{th} response. The response resolver also has a look-ahead feature that checks all the search flags in parallel for regions of no response. If a no-response region is found, the counting resolver will automatically step over that region. This check can speed the finding of responses considerably.

The response resolver uses an 8-bit up/down counter (CAM ADDR REG) to generate the address of an interrogation response or an available space cell. This counter is incremented or decremented depending upon whether the search is to proceed upward or downward. Each time the response flag or the available space flag corresponding to the cell address in the counter is set, an 8-bit INDEX counter is decremented. When the value in the INDEX counter becomes 0, the clock control stops the 5-MHz clock signal of the address counter (CAM ADDR REG) and the count in the address counter represents the cell address of the desired response or available space. The INDEX counter must have already been loaded with the value "i" for the i^{th} item before the 5-MHz signal can be applied to the address counter for address calculation. If the address counter overflows (counter exceeds $2^8 - 1$ while incrementing or goes below 0 while decrementing), a fail is generated.

The logic labeled SELECTOR in Figure 2 is used to determine which set of flags the system is to examine. If FINDMT is "true," the response flags will be examined; if FINDMT is "false," the available space flags will be examined.

The HIT INDICATOR logic generates a pulse, HIT, when the flag corresponding to the address in the address register is set. The address register is incremented or decremented through each address in sequence unless the look-ahead logic has indicated that a block of flags have been reset and may thus be bypassed.

The logic labeled NOHIT CHECK is used to indicate that a block of flags or all of the flags have been RESET:

$\overline{HIT}^{0}_{255}$	All the flags are RESET.
$\overline{HIT}^{0}_{127}$	All flags representing cells 0 through 127 are reset.
$\overline{HIT}^{127}_{255}$	All flags representing cells 128 through 255 are reset.
NOHIT	The 16 flags about to be examined are all reset.

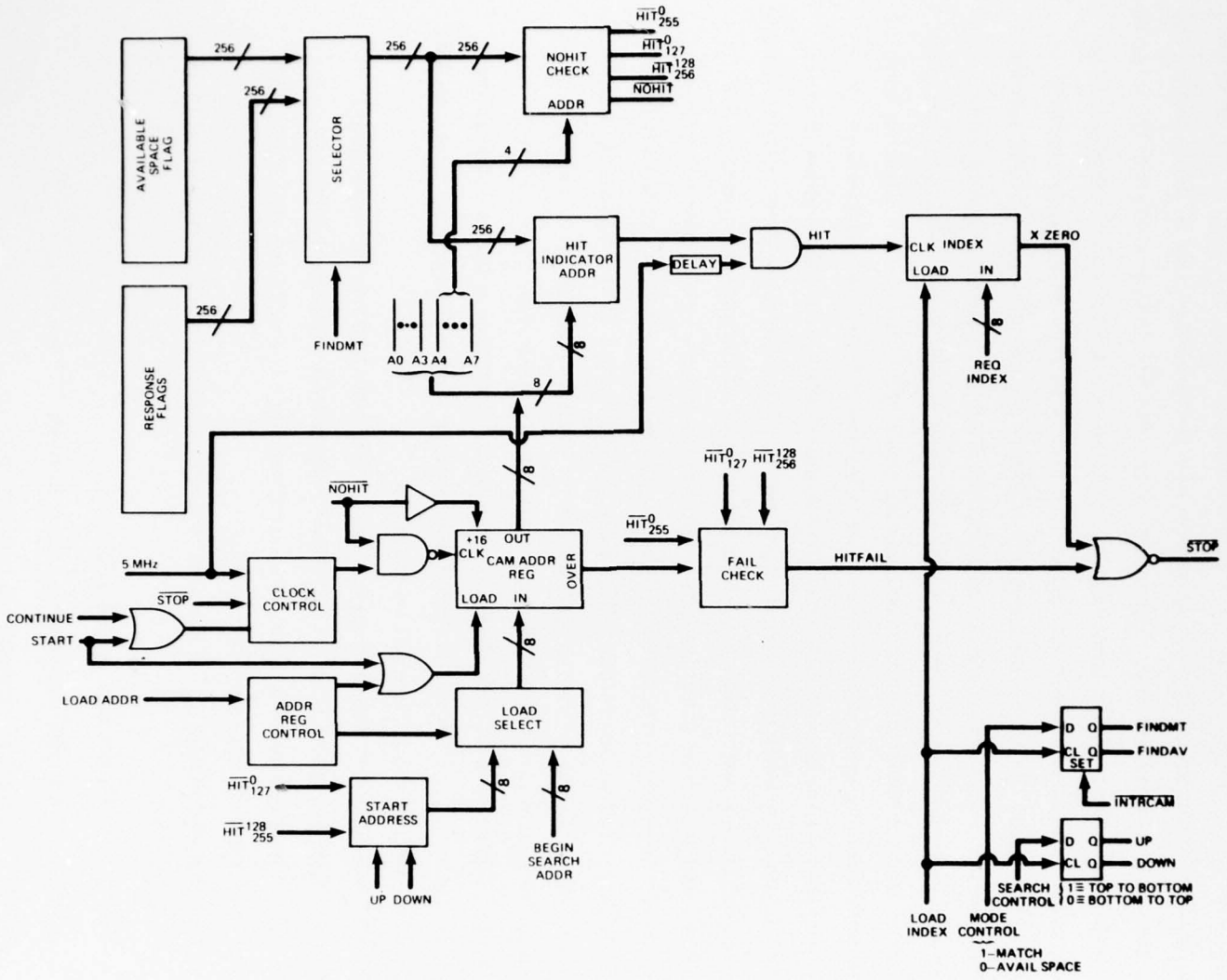


Figure 2 – Counting Response Resolver

These look-ahead checks allow the address counter to skip over those blocks of flags that have all been reset, thus reducing considerably the average time used to calculate a flag address.

WORKING MEMORY

Working memory consists of 256 cells of storage. Each cell consists of 48 data bits and 8 page bits. Figure 3 is a block diagram of the memory. The 48 data bits of each cell are in CAM. The 8 page bits are in RAM.

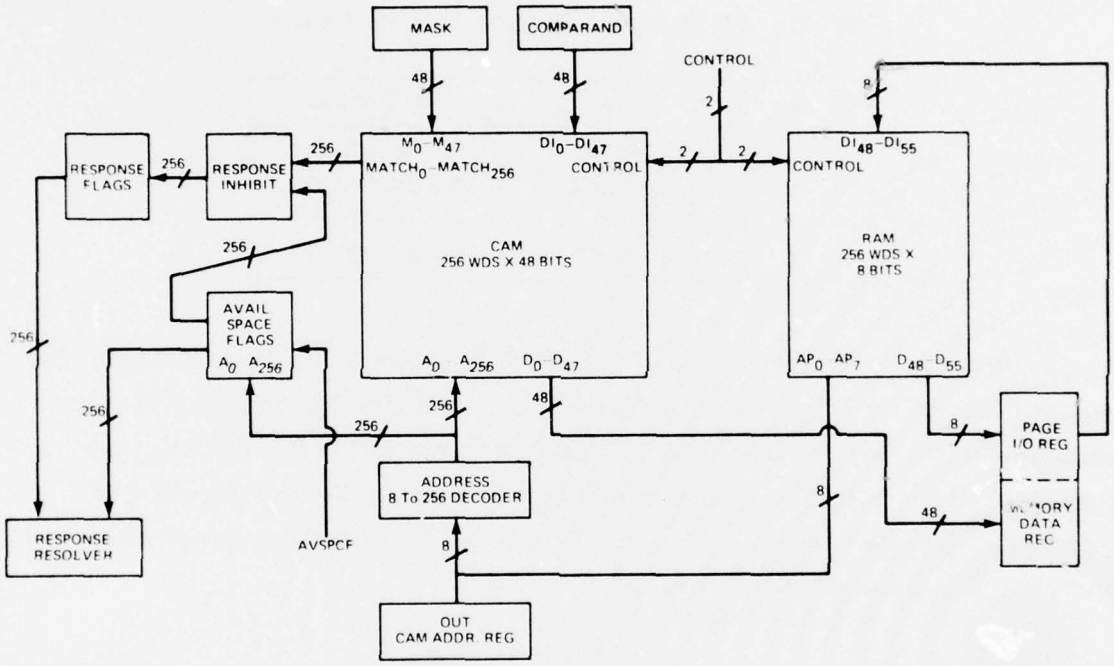


Figure 3 – Working Memory

Inputs to the memory are the 48 bits from the mask register, M_0-M_{47}; the 48 bits from the comparand, DI_0-DI_{47}; 256 address bits, A_0-A_{256}; and two control bits. The comparand register is also the input register for interrogation. Information in the register is matched against data stored in memory. Bits are masked out of the interrogation by the mask inputs, M_0-M_{47}. A logic "0" for M_i allows the i^{th} data bit to be recognized. A logic "1" causes the data bit to be ignored. Memory accesses to cells are made as indicated by the 256 address lines, one address line for each cell. A logic "1" on one of these lines selects a cell location for the access. Only one of the address inputs may be a logic "1" during a read access, but several may be logic "1" for a multiple write. The write access is a mask operation in that

information in selected bits is changed during a write cycle. The signals for the 256 address lines are generated by an 8-to-256 decoder which uses the 8-bit binary address in the address register as input. This 8-bit address goes directly to the RAM because the RAM continues its own internal address decoder. This address register is the counting register used in the multiple response resolver. If the multiple response resolve implementation becomes a ripple-carry method or a tree-structured logic, then the 256 address lines will be generated directly and a 256-to-8 address encoder must be added for the RAM. The 2-bit control line determines the memory operation. Table 1 shows the memory control truth table.

TABLE 1 – MEMORY CONTROL TRUTH TABLE

Control	CAM Operation	RAM Operation
0 0	WRITE	WRITE
0 1	INTERROGATE	NO OPERATION
1 0	READ	READ
1 1	NO OPERATION	NO OPERATION

Outputs from the memory consist of 256 match indicators and 56-bit data output. The 56-bit data output consists of 48 bits output from the CAM (D_0–D_{47}) and the 8-bit page output from the RAM (D_{48}–D_{56}). The 256 match indicators ($MATCH_0$–$MATCH_{255}$) are used to set the response flags. The response inhibit logic is needed to forestall the setting of response flags for cells whose available space flag is logic "1", since those cells are empty and should not be included in any interrogation.

MASK PROCESSOR

Figure 4 shows the mask logic and its connections to the CAM. Masks determine which bits of each CAM cell are used for comparison during interrogation. For each set bit of the mask that has the corresponding bit of a memory cell and the comprehand register the same, the response flag for that memory cell is set. The mask logic consists of a mask memory and a mask address register.

The mask memory allows the user to define four masks, any one of which may be used as the interrogation mask. Once a mask has been selected for use as the interrogation mask, its use will continue for all successive interrogations until a different mask is selected. The user may redefine any of the four masks at any time by specifying the mask address and the new mask to be stored.

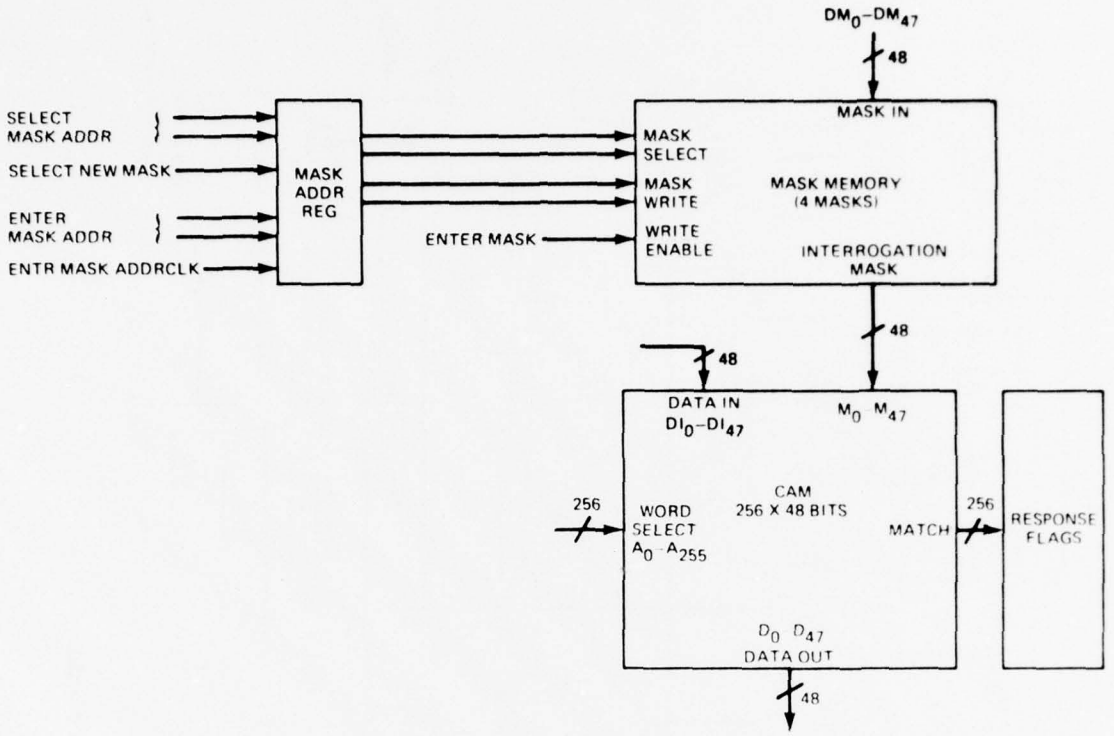


Figure 4 – Mask Processor

Masks are accessed by addresses stored in the mask address register. The mask address register contains two 2-bit addresses, one for entering a new mask into the mask memory and the other for selecting a mask as the interrogation mask. To enter a new mask, the following signals must occur, in the order listed:

- Address of desired mask on ENTERMASKADDR lines
- Pulse signal on ENTRMASKADDRCLK line
- New mask on MASKIN lines
- Pulse signal on ENTERMASK line

A mask is selected as the interrogation mask with the following signals:

- Address of desired mask on SELECTMASKADDR lines
- Pulse signal on SELECTNEWMASK line

PAGED MEMORY

Paged memory, Figure 5, swaps the contents of the CAM with one page of the CCD memory. This swap takes place automatically whenever an operation (insertion, deletion,

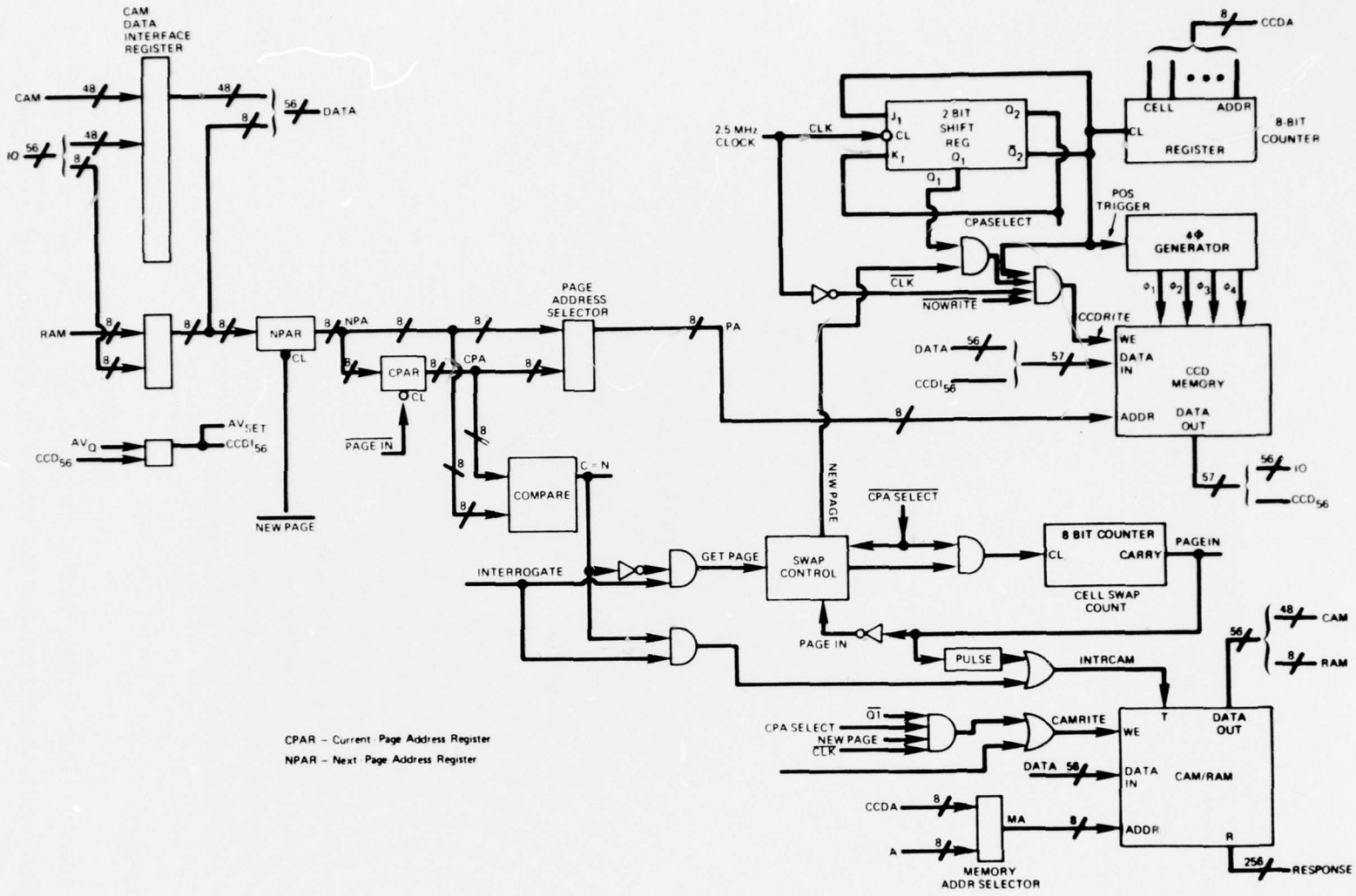


Figure 5 – Paged Memory

retrieval) is performed and there is a difference in the contents of the next-page address register (NPAR) and the current-page address register (CPAR). The next-page address register receives data from bits 48-55 of the CAM DATA INTERFACE register. The CAM DATA INTERFACE register receives data from either the 56 bits (48 bits CAM, 8 bits RAM) of the last word in the CAM referenced by an operation or the 56 bits as loaded from the I/O bus. After the new page has been swapped in and the current page has been swapped out, the contents of the NPAR are loaded with the contents of the current-page address register.

The CCD memory (Figure 6) is composed of 14592 shift registers, each containing 256 bits. The shift registers are arranged in 256 pages. Each page has 57 parallel shift registers, one for each bit in a CAM cell (1 available space flag, 48 data bits, and 8 page bits). The 256 bit length of each shift register is the number of cells in the CAM, thus one complete cycle of 256 shifts will fill the entire CAM. Each of the 256 pages are addressable.

The components proposed for the CCD will allow any two pages to be accessed between shifts of the registers.

This multiple accession allows the contents of each CAM cell to be swapped with the corresponding CCD memory word during one shift of the CCD memory. Thus, during a complete cycle of 256 shifts, the entire CAM may be swapped with the page in the CCD memory. The NPAR (New Page Address Register) and the CPAR (Current Page Address Register) are used alternately as the Page Address (PA) under control of the PAGE ADDRESS SELECTOR.

The CCD memory requires a four phase clock. During the four phases, two shifts are performed. Thus, the 4Φ GENERATOR (Figure 5) is designed to generate all four phases, two phases at a time. After each two-phase generation, a read/write access is made to swap one cell between CAM and CCD. The output (CPASELECT) of a 2-bit shift register is used as a cue for generating each set of two phases. Additionally, this shift-register output drives the word address register, which is an 8-bit counter. This register is used to keep track of the cell address represented by each shift of the CCD.

The swap control logic insures that the CCD and CAM are synchronized before data swapping is initiated. A cell swap count is kept using an 8-bit counter. When the number of cells transferred is equal to 256, the swap control stops the swap operation and the system can continue with the requested user operation.

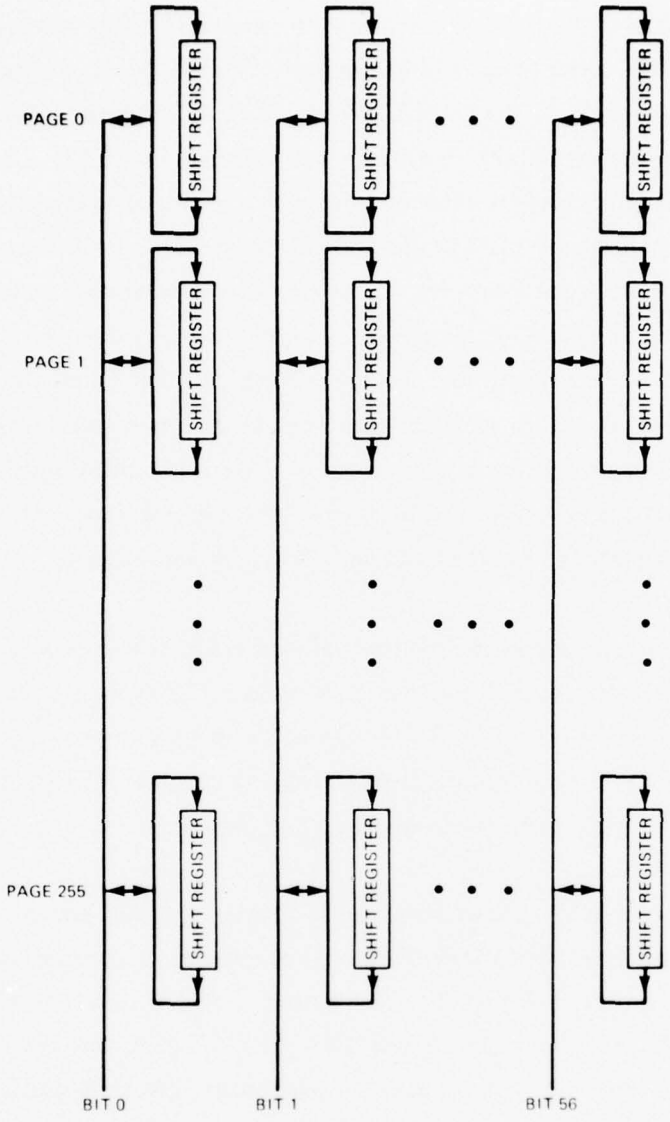


Figure 6 – CCD Memory

MICROPROCESSOR

The microprocessor is the associative memory control. It receives requests from the user. Each user request is decoded into a command sequence for the associative processor logic networks. In generating the command sequence, the microprocessor issues commands one at a time and waits for an acknowledgment of completion before issuing the next command in the sequence. When the entire command sequence has been completed, the results are sent to the user. This string of commands is the program for the microprocessor. After a program

has been tested and throughly debugged, it can be implemented using read only memory (ROM) for any future implementations of the associative memory.

The programmable controller provides flexibility in allowing new user requests to be added or current user requests to be modified. These requests may be either specific, for a particular task, or general for many tasks.

The associative memory is designed to execute requests that are generated as a result of GIRL statements in a host computer. These requests include insertion, deletion, and retrieval. Each of these requests has two modes, normal and indexed. The normal mode is intended for use in processing requests involving single-value lists (single response). Normal user requests operating on multivalue lists (multiple response) are legitimate. A normal retrieval will return the first item (response), a normal deletion will clear the entire list, and a normal insertion will place a new item at the end of the list. Indexed mode operations are intended to operate on lists. These operations retrieve, delete, insert, or replace the i^{th} item on a list. If a user request cannot be successfully completed, i.e., no response to a retrieval request, a failure flag is set. This flag is sent to the user as notification that, given the current state of the CAM, the request cannot be completed.

The insertion, retrieval, and deletion operations will be implemented in the associative memory controller. Initially these operations will be performed sequentially, one request at a time. Modifications or additions to the operational capabilities may be added as needed. These additions may be complex requests to simplify user work. Operational modifications may include adding a request stacking capability which would allow the user to make a string of requests and get back the final result without intermediate results. This request stacking could be best implemented if the associative memory had a direct memory access (DMA) to the host computer. The request stacking would increase operational speed and reduce the amount of data (requests and results) that would have to be transferred between the associative memory and the host computer.

CONTROL SOFTWARE

Introduction

The following pages describe the six microprocessor control programs: Executive, Retrieval, Deletion, Destructive Insertion, and Mask Handler. Preliminary flow charts for each are provided. The symbols used are indicated in Figure 7. Tables 2 through 5 describe the mnemonics used in the flow charts to indicate program entry points (Table 2), subroutines (Table 3), program storage (Table 4), and hardware signals (Table 5).

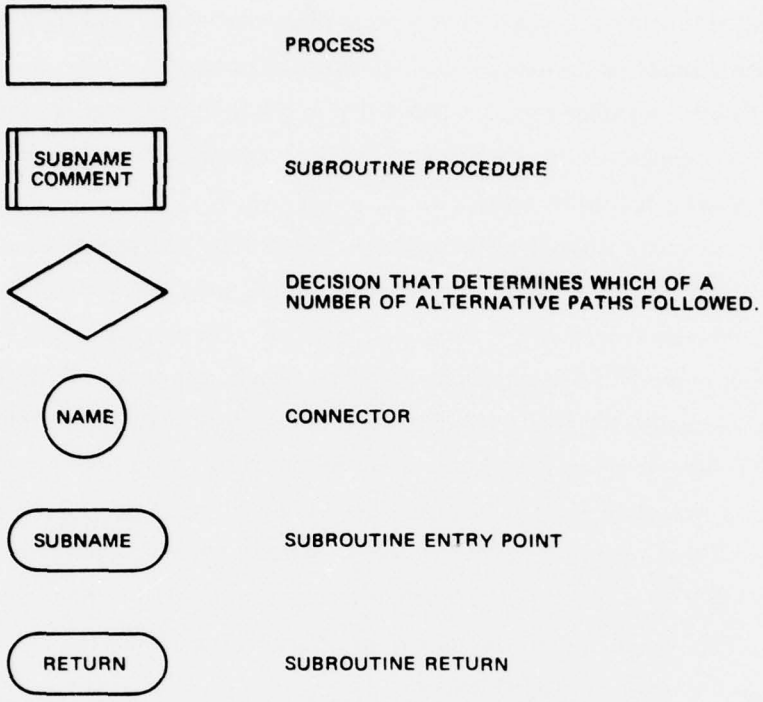


Figure 7 – Flow Chart Symbols

TABLE 2 – CONTROL SOFTWARE ENTRY POINTS

Entry Points	Description
COMOUT	Entered to send COMPARAND to host, and then set fail condition.
DELET	Entered to perform normal deletions.
DNSRT	Entered to perform destructive insertions.
EXIT	Entered to reset busy flag and to return to START.
FAILR	Entered to set hardware fail flag, and then go to EXIT.
INSERT	Entered to transfer COMPARAND to CAM cell.
INSRT	Entered to perform insertions.
MASK	Entered to obtain new interrogation mask from host
NDELT	Entered to perform next deletions.
NOCELL	Entered to set NOCELL flag, and then set fail condition.
NRFTR	Entered to perform next retrievals.
OUTCAM	Entered to send CAM cell to host, and then go to EXIT.
RETR	Entered to perform indexed and normal retrievals.
RSTFLG	Entered to reset AVSPC flag for a CAM cell.
START	Entered to start the Executive program.
XDEL	Entered to perform indexed deletions.

TABLE 3 – CONTROL UTILITY SUBROUTINES

Subroutine	Description
NEXT	Generates hardware signals for finding the CAM cell address of the next response of the previous interrogation.
RETRIEVAL	Generates hardware signals for finding address of the I^{th} CAM cell responding to the previous interrogation.
AVSPC	Generates hardware signals for finding the first CAM cell whose corresponding AVSPCF is set.
NXAVSPC	Generates hardware signals for finding the next responding CAM cell from the cell whose address is in the CAM ADDR REG.

TABLE 4 – PROGRAM LOCATION MNEMONICS

Control Program Locations	Description
INDEX	Used to save the Index of the operation requested by the host.
SAVE 1 SAVE 2 SAVE 3	Locations used during insertions to save and restore the shuffled items on their way from and to the CAM ADDR REG; also used to store the mask address.
SAVINX	Location to save the requested index so as to restore and/or change the index sign during special cases of insertions.
TEMP	Location to save contents of CAM cell during insertions which require shuffling of CAM item.
UDSAVE	Location to save original direction of search.

TABLE 5 – HARDWARE MNEMONICS

Hardware Signals	Description
AVSPCF	Available space flag for the CAM cell addressed by the CAM ADDR REG.
BEG SEARCH ADDR	Input for the CAM ADDR REG.
BUSY	Flag which when set indicates to the host that a request is being processed.
CAM	CAM cell addressed by the CAM ADDR REG.
COMPARAND	Comparand register for the CAM.
CONTINUE	Clocked signal that starts the hardware response resolver search from the CURRENT ADDRESS in the CAM ADDR REG.
$DM_0, DM_1, \ldots, DM_{47}$	Data input for the mask memory.
ENTER MASK	Clock signal that causes $DM_0, DM_1, \ldots, DM_{47}$ to be transferred into the mask memory location indicated by the mask address register.
ENTER MASK ADDR	Two-bit input to the mask address register used for the mask write address of the mask memory.
ENTR MASK ADDR CLK	Clock signal that causes the address, ENTER MASK ADDR to be transferred into the mask address register used for the write address of the mask memory.
FAIL	Flag which when set indicates to the host that a request could not be completed.
LOAD INDEX	Clock signal that causes the index register in the response resolver to be loaded with REQ INDEX.
MODE	Response resolver mode control; when set causes match searches, and when reset causes available space searches.
NOCELL	Flag which when set indicates to the host that the CAM has no empty cells.
REQ INDEX	Input line for the response resolver INDEX REGISTER.
SEARCH	Response resolver search control flag; when set causes top-to-bottom search, and when reset causes bottom-to-top search.
SELECT MASK ADDR	Input to the mask address register which causes the addressed mask to be read from the mask memory.
SELECT NEW MASK	Clock signal that causes the address SELECT MASK ADDR to be transferred into the mask address register for the read address of the mask memory.
START	Clock signal that starts the response resolver search, either from the top or the bottom of the CAM depending on the state of search.
STOP	Signal that indicates the completion of a response search.

Executive

The Executive program reads commands from the host computer, interprets the commands, transfers control to the appropriate handler, and returns the results to the host. The flow chart for the Executive is shown in Figure 8. This program loops on START, waiting for a command from the host. When a command is available a busy flag is set and the command is read from the host. The failure flag is reset.

If the command is a GIRL type, the different classes of commands are checked. If the GIRL command is of the NEXT class, a memory interrogation is unnecessary, since memory interrogation results from a previous command can be used. If this NEXT class operation is also INDEXED, the index has to be read from the host. Control is transferred to the appropriate handler according to the type of command.

If the GIRL command is not of the NEXT class, a memory interrogation must be made. First the comparand must be read from the host, then the memory interrogation is made. If the command is INDEXED, the index is read. Control is transferred to the appropriate handler according to the type of the command.

Non-GIRL commands perform mask-manipulation operations. One such operation performs the selection of a new mask. As many as four masks, one of which will be the interrogation mask, may be stored in the mask memory. Each mask can be updated. Updating does not affect the mask select address.

EXIT, OUTCAM, COMOUT, and FAILR are Executive entry points for returns from the handlers. EXIT resets the BUSY flag and transfers control to START to await a new command. OUTCAM sends the contents of the current CAM cell to the host and then transfers control to EXIT. COMOUT sends the output of the comparand register to the host and then transfers control to FAILR.

Retrieval

Figure 9 illustrates the flow of control through the retrieval handler. The retrieval handler has two entry points, NRETR and RETR. NRETR is the entry point for NEXT class operations. The NEXT utility subroutine is called to find the i^{th} response from the previous retrieval. For retrievals other than the NEXT class, the entry point is RETR. The RETRIEVAL utility is called to find the i^{th} item on a list. After a return is made from either of these utilities, the failure flag is checked to see if the retrieval was successfully performed. If the retrieval was not performed, control is returned to the Executive via COMOUT to output the contents of the comparand register and to set the failure flag.

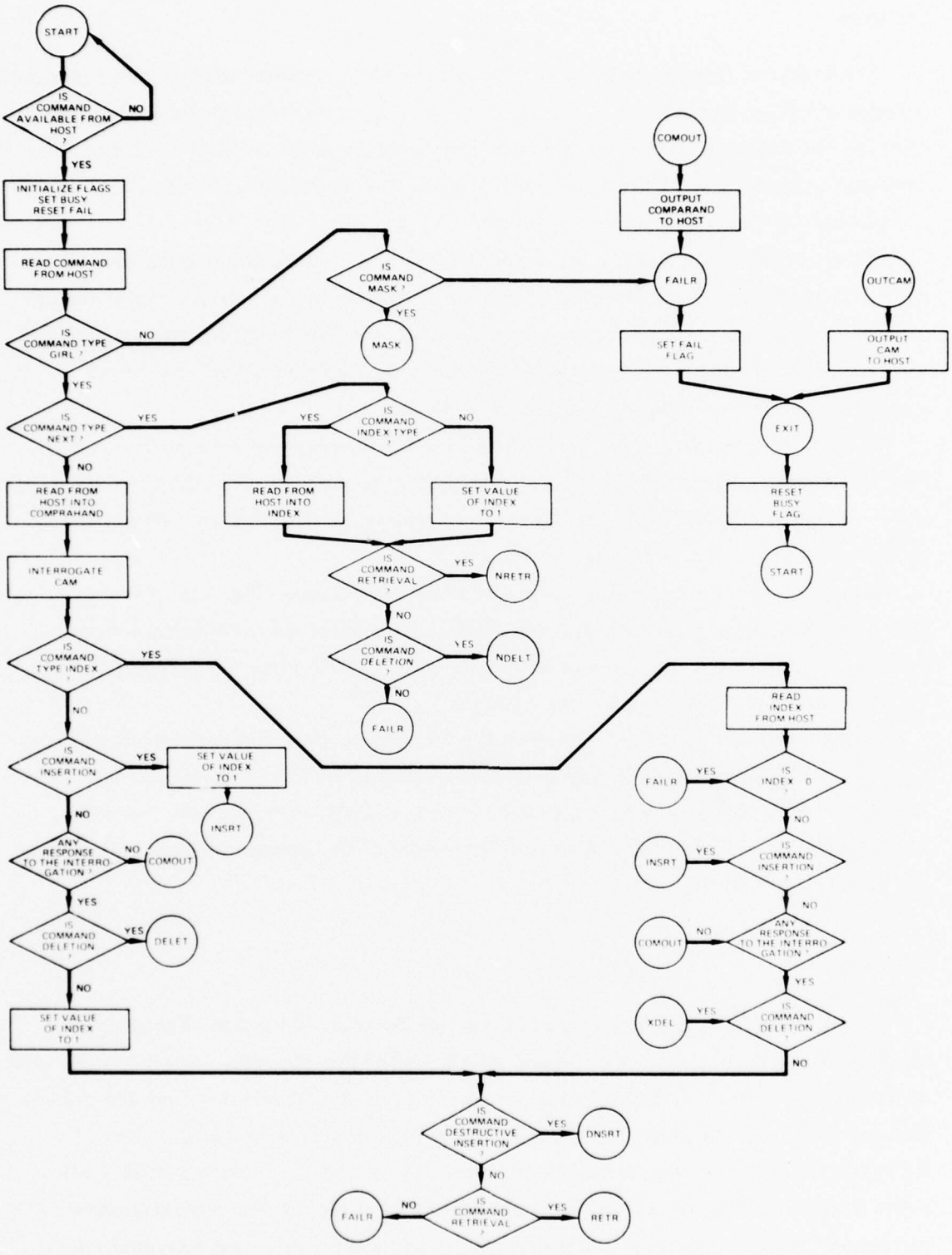


Figure 8 – Executive Flow Chart

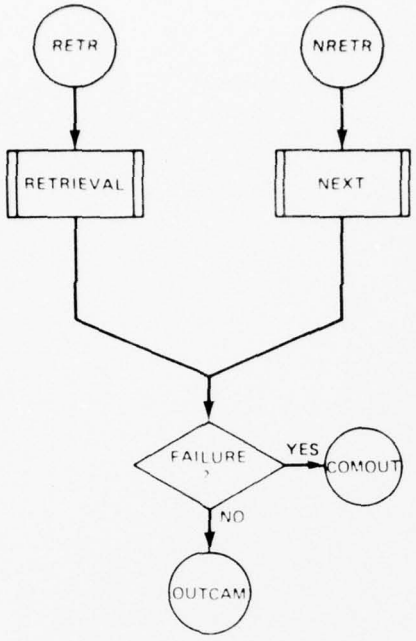


Figure 9 – Retrieval Flow Chart

If the retrieval was successful, return is made to the Executive via OUTCAM to send the host the retrieval results.

Figure 10 indicates the flow of control in the utilities NEXT and RETRIEVAL. The two utilities are basically the same except that in the RETRIEVAL utility the clock START initializes the CAM ADDR REG to 0 for up-retrievals and to 255 for down-retrievals, whereas in the NEXT utility the clock CONTINUE causes the retrieval to begin at the address in the CAM ADDR REG. In both utilities the mode control is set for retrieval, the search control is set for either an upward or a downward search (according to the sign of INDEX), and the absolute value of INDEX in the REQ INDEX register of the CAM is set to the index value requested by the user.

Deletion

The flow diagram for the deletion handler is provided in Figure 11. The deletion handler has three entry points. DELET is the entry point normally used when the entire list that responded to the interrogation is to be deleted. To delete an item (cell) in the CAM, the available space flag for that cell is set. The first item to respond is output to the host.

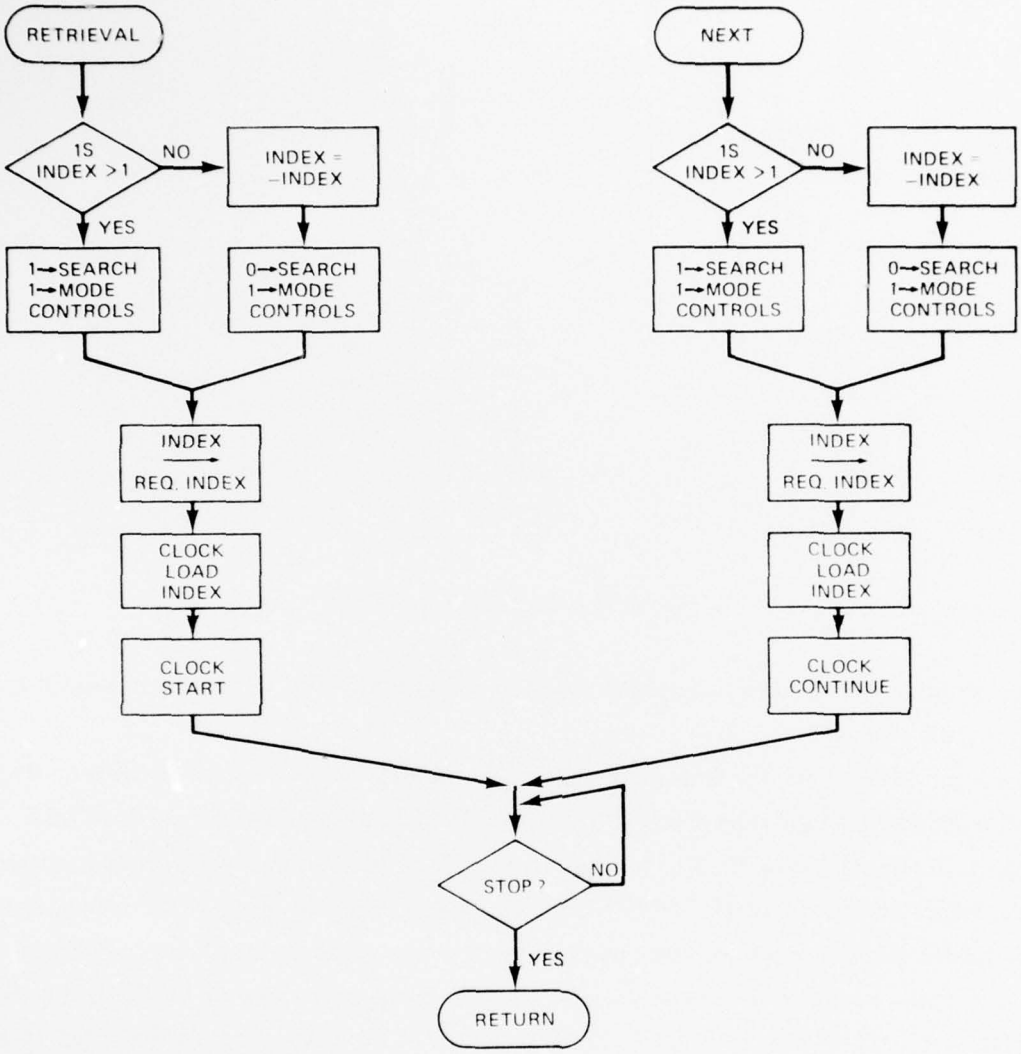


Figure 10 – The NEXT and RETRIEVAL Utilities

XDEL is the indexed deletion entry point when a single item is deleted from a list. **INDEX** contains the signed number indicating which item on the list is to be deleted. **NDELT** is the delete-next-item entry point. The i^{th} item from the previous response is deleted from the CAM. In all cases the first item deleted is sent back to the host.

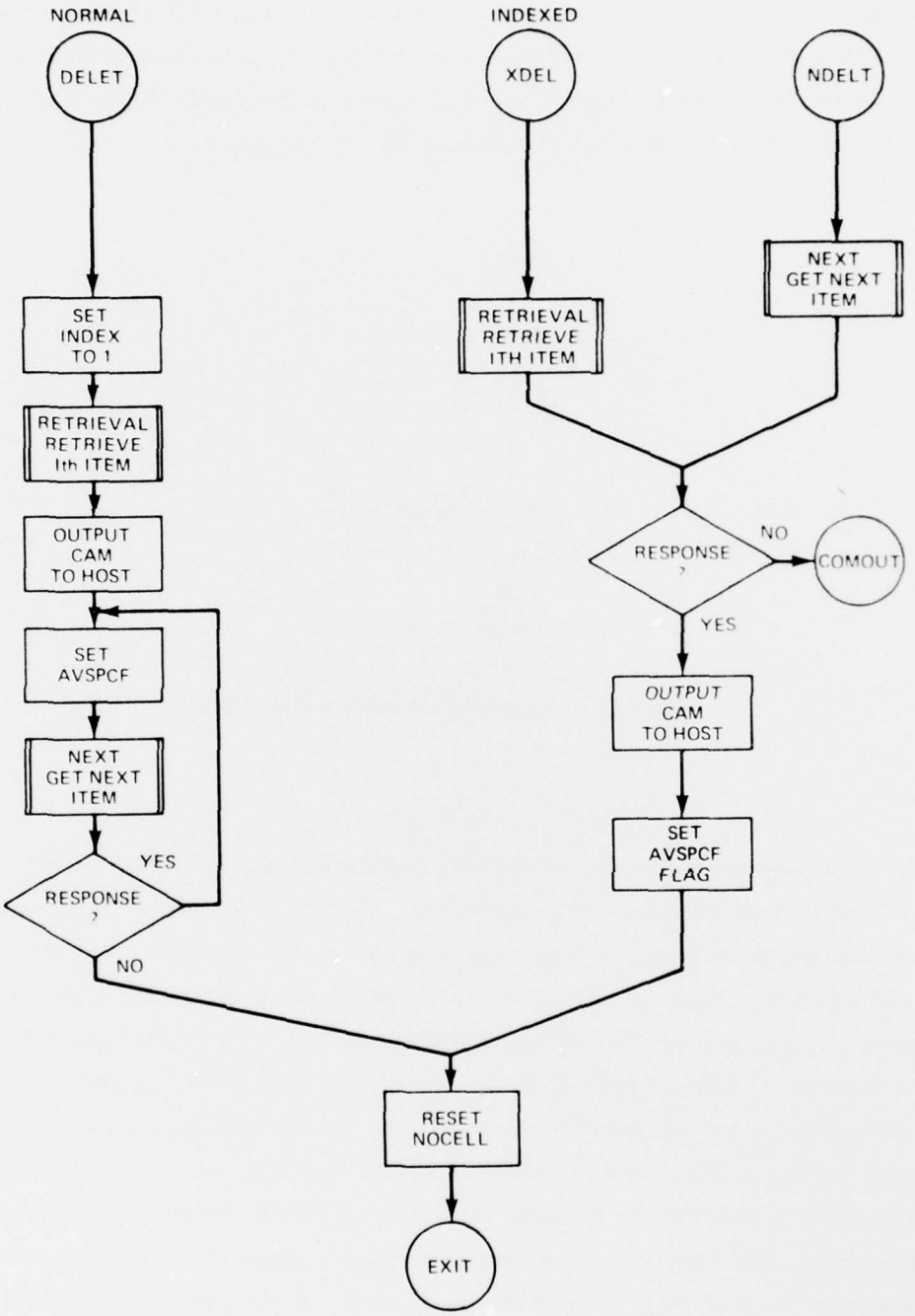


Figure 11 – Deletion Flow Chart

Destructive Insertion

The destructive insertion handler replaces a specified item in the CAM with the desired item. The flow diagram for this handler is given in Figure 12. The entry point is DNSRT. First the item to be replaced is located and then control is transferred to a routine that writes the comparand into the CAM, thus destroying the original item.

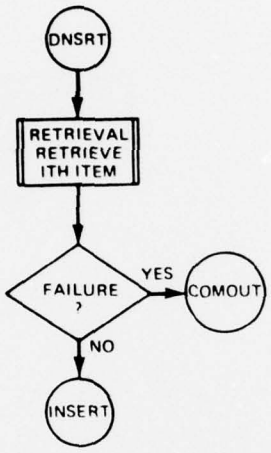


Figure 12 – Destructive Insertion Flow Chart

Insertion

The flow diagram for the insertion handler is given in Figure 13. The handler entry point is INSRT. Insertion places an item in CAM. Items may be inserted anyplace on a list. Currently, the list position of any item responding to a particular interrogation is indicated by its relative position in memory with respect to the set of items responding to the particular interrogation. Items can be inserted only into those available cells having their AVSPCF flags set. To insert an i^{th} item on the list, the handler retrieves the $i^{th}-1$ item and then tries to find an available cell beyond the $i^{th}-1$ item. If the available cell exists, a check is made to see if there are any response items located between the available cell and $i^{th}-1$ item. If not, the desired item is stored in the available cell and the AVSPCF flag for that cell is reset. If such intervening items exist they are moved, their relative positions being maintained, until sufficient room has been made to store the desired item. If there is no available cell available following the $i^{th}-1$ cell, the handler tries to find an available cell in front of the $i^{th}-1$ cell, and, if successful, moves items, maintaining relative order, until space is made in the i^{th} position for the desired item. If the command is to insert the i^{th} item and there are fewer

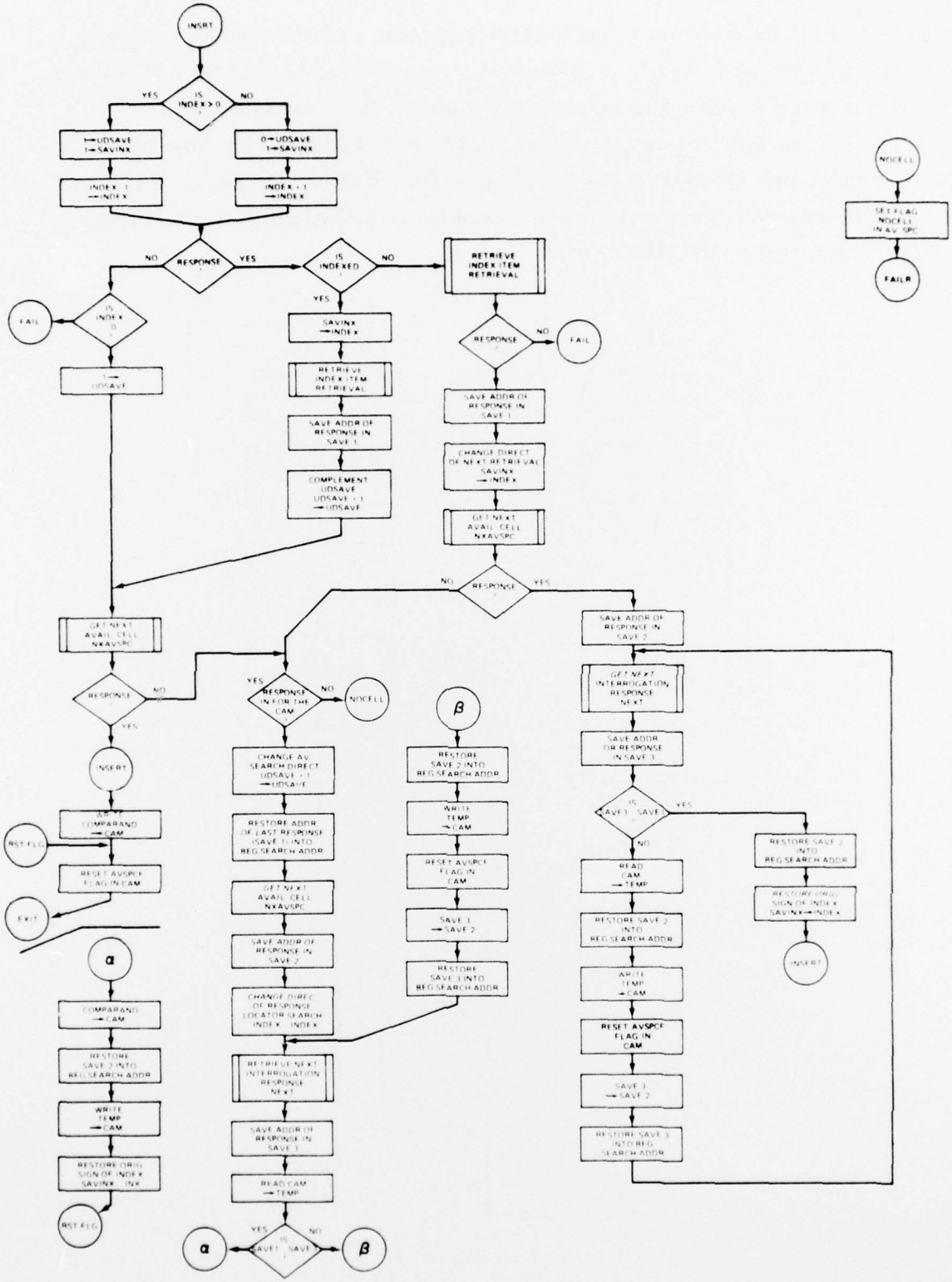


Figure 13 – Insertion Flow Chart

than I-1 items in the CAM responding to the interrogation, a failure condition exists and control is transferred to FAILR. If an insertion is requested and none of the AVSPCF flags are set, the NOCELL flag will be set, and control will be transferred to FAILR.

The insertion handler calls two utilities, AVSPC and NXAVSPC. The flow diagram is given in Figure 14. AVSPC finds the first or last cell in AVSPC, depending upon UDSAVE. NXAVSPC finds the cell nearest the cell addressed by the CAMADDRREG, either ahead or behind, as indicated by UDSAVE.

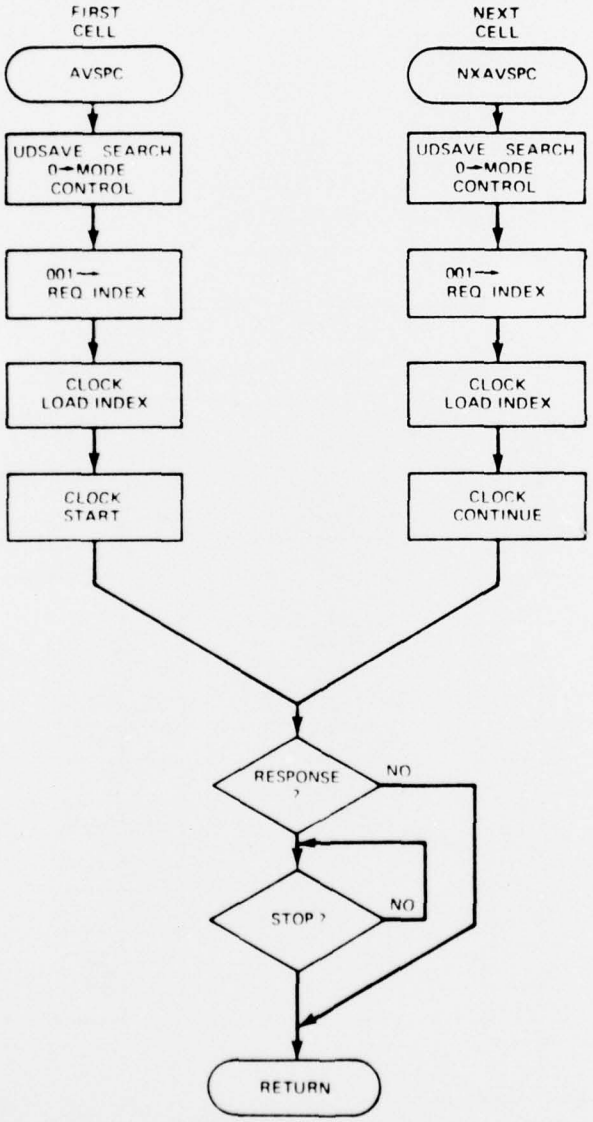


Figure 14 – Insertion Utilities

Mask Handler

The Mask Handler, Figure 15, is used to enter a new mask from the host into the mask memory and/or to select one of the four masks stored in the mask memory as the interrogation mask. The handler first obtains the address (0-3) of the mask register to be accessed. If a new mask is being defined, the mask address is stored into the mask address register by placing the address on the ENTERMASK ADDR lines and generating a clock pulse on ENTRMASKADDRCLK. Next the new mask is obtained from the host and entered on data lines $DM_0, DM_1, \ldots, DM_{47}$. A clock pulse is generated on ENTERMASK causing the new mask to be stored in the desired mask memory location. With the new mask defined, the handler determines whether this mask is to be selected as the new interrogation mask. If not, control is returned to the Executive via EXIT. To select one of the stored masks as the interrogation mask, the mask address is stored in the mask address register by placing the address on the SELECTMASKADDR lines and generating a clock pulse on SELECT NEW MASK. With the mask selected, the handler returns control to the Executive via EXIT.

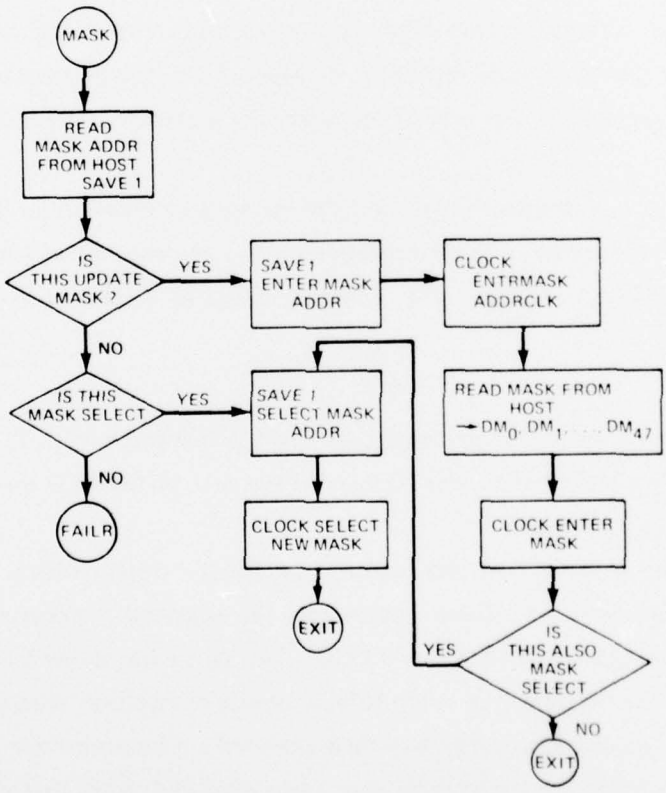


Figure 15 – Mask Handler Flow Chart

SYSTEM COST

COST ANALYSIS

Several CAM-CCD memory-mix options are set forth, along with the resulting cost and performance tradeoffs. The implementation cost for each is plotted against the average retrieval time so that a design engineer may determine the most cost effective implementation for his specific application.

The average retrieval time depends upon the number of times the pages have to be swapped between CAM and the CCD mass memory. As the page size increases (more CAM), the average number of page swaps decreases. In fact, doubling the page size should reduce the average number of page swaps at least by half. The average retrieval time can be determined as soon as the number of retrievals per page swap is known.

The average number of page swaps for any particular implementation is dependent upon the user's application and how well the user partitions the information into the pages. The expected use of the associative memory system is to process directed graph structures[1] which are constructed with node-link-node triplets. Graphs that are tree structured (no loops) present a worst case as regards retrievals per page swap, since circuits are generally stored on a single page which results in more retrievals per page. Thus, the performance as measured in terms of retrievals per page is dependent both upon the memory configuration and upon the application.

The system cost is dependent only upon the memory configuration. The curve shown in Figure 16 reflects the increase in implementation cost as the amount of CAM increases for a system that has a 65K-cell CCD memory. The percentage of CAM is computed as

$$\frac{\text{CAM}}{65\text{K}} \times 100$$

For the maximum case where the percentage of CAM is 100 percent, CCD memory is unnecessary. The low end of the curve flattens to the cost of the CCD memory, the microprocessor, and the control logic.

The relationship between cost and performance for the system with a 65K-cell CCD memory is shown in Figure 17. Curve 2 represents the expected performance of the acoustic transient classifier using a mix of CAM and CCD. This curve was derived by generating a paging strategy for the decision tree taken from a similar recognition scheme designed for another problem. The paging strategy was then analyzed to determine the expected performance by first calculating the retrieval-per-page swap ratio and, from that result, determining the average retrieval time. The directed graph for the acoustic transient classifier will have

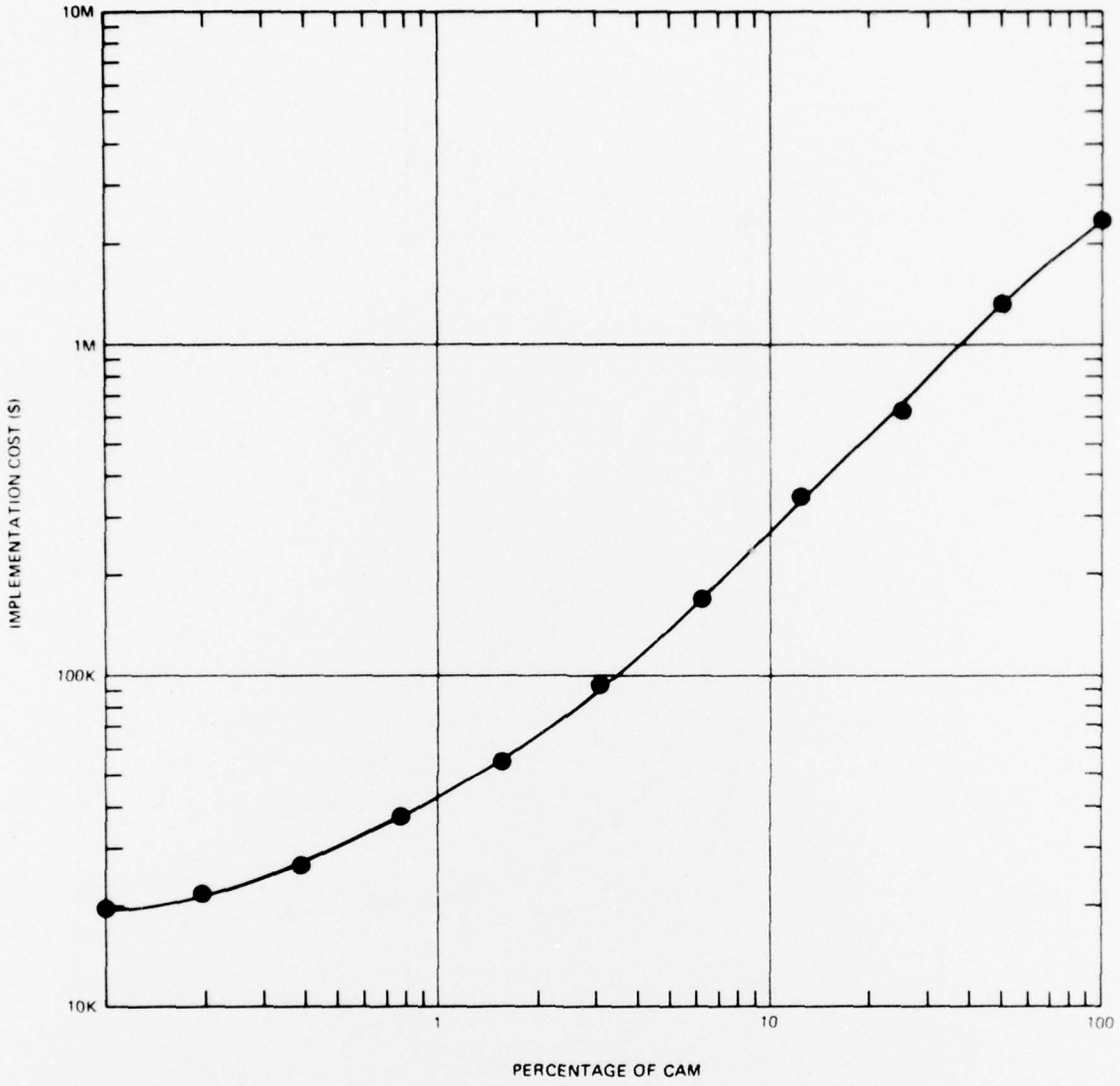


Figure 16 – Implementation Cost versus CAM Memory Percentage

many similarities to the directed graph of the recognition scheme used for the calculation.

Table 6 tabulates the system performance data shown in the figures. The average number of retrievals per page is used to determine average retrieval time. For example, consider a system comprising 65,536 words of CCD memory with 256 words of CAM (0.39%). The average number of retrievals per page is estimated to be 16. The time required for one swap is 450 microseconds. The average retrieval time with no page swaps is 9.2 microseconds. Thus for the described system the average retrieval time with page swaps is

$$9.2 + 450/16 = 37.3\ \mu\text{sec}$$

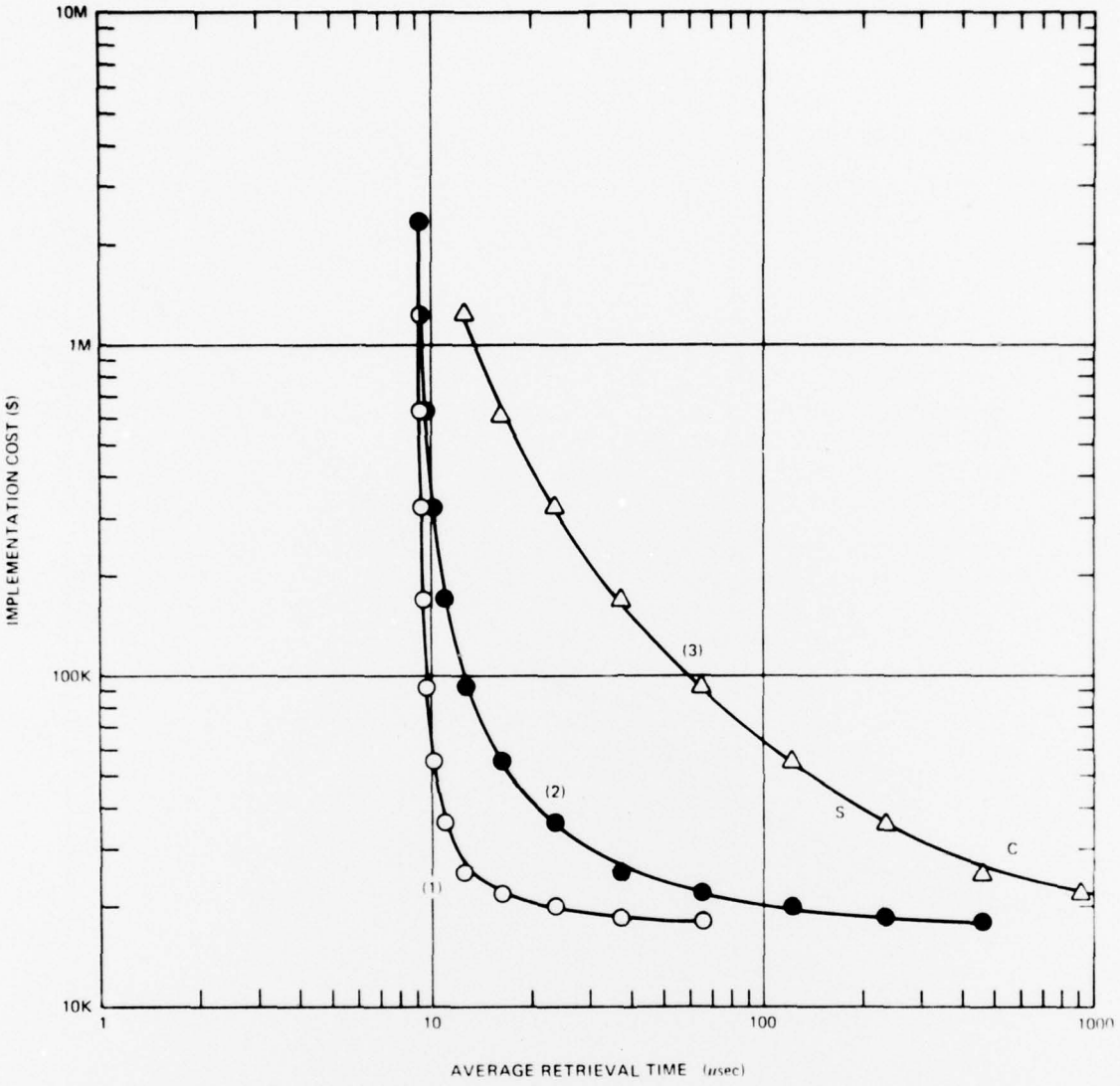


Figure 17 – Cost Performance Relationship

Curves 1 and 3 in Figure 17 are used to illustrate what would result if the actual number of retrievals per page differed from the estimate. Curve 1 represents the situation in which the actual number of retrievals per page is eight times that estimated. In the example involving 256 words of CAM, the average retrieval time would be 12.7 microseconds. Curve 3 represents the performance if the actual number of retrievals per page is one-eighth of that estimated. This situation would result in an average retrieval time of 234.2 microseconds for the example.

TABLE 6 – ASSOCIATIVE MEMORY PERFORMANCE

Cost ($K)	CAM Page Size /percent	Average Retrievals Per Page	Average Retrieval Time (μsec)
2340.0	65,536/100	∞	9.2
1230.0	32,768/50	2048	9.3
623.0	16,384/25	1024	9.6
322.0	8,192/12.5	512	10.1
169.0	4,096/6.25	256	11.0
92.7	2,048/3.13	128	12.7
54.7	1,024/1.56	64	16.2
36.0	512/0.78	32	23.3
25.5	256/0.39	16	37.3
22.0	128/0.19	8	65.5
19.8	64/0.093	4	121.7
18.5	32/0.046	2	234.2
18.0	16/0.023	1	459.2

The most cost effective implementation is in the region of the bend or elbow of the curves where the costs range from $25K to $55K. In this region the performance per dollar is maximum. In areas where the costs exceed $55K, the cost increases greatly for small increments in performance. In areas where the cost is below $25K, the performance decreases sharply for small decrements in costs. Physical constraints, both budgetary and operational, may dictate the adoption of an implementation that is not the most cost effective.

Implementation of the system proposed in this report is in the cost effective region. This particular implementation was selected because it was ideally suited for current integrated circuit implementation and provided good operational capability and acceptable speed for implementing practical problems. The 256 word CAM requires 8-bit binary address registers which are easy to implement using integrated circuit 4- or 8-bit registers. The 256 pages also require an 8-bit binary address, meaning that page address registers, too, can be implemented using 4- or 8-bit integrated circuit registers.

IMPLEMENTATION COST

The cost of a system is a function of the cost of the circuit components, the cost of the supporting circuitry and packaging, and the cost of interconnection cables. The quantity and cost of several factors have been estimated. The estimated cost of the proposed system is $25,500 (Table 7). The following paragraphs discuss some of these cost factors.

TABLE 7 – IMPLEMENTATION COST

Description	Quantity	Power (Watts)	Procurement Cost ($)	Pin Cost ($)
16K-bit CCD memory	228	38.1	5,574.60	1,231.20
64-bit CAM	194	96.0	4,000.00	2,764.80
256-bit RAM	8	4.0	134.40	38.40
Microprocessor controller	1	7.2	700.00	350.00
TTL logic IC's	1000	56.5	3,000.00	5,000.00
Cable connectors	100		100.00	500.00
Cabinets and hardware			2,500.00	
			16,009.00	9,484.40
Total Cost:			25,493.40	

Circuit Component Cost

Most of the circuit components for the system are semiconductor integrated circuits. These integrated circuits include the memory, the microprocessor, and the interface and control logic.

Three types of memory integrated circuits are required – a CCD, a CAM, and a RAM.

- The paged memory uses the 16-bit Intel 2416 integrated circuit. This integrated circuit costs $24.45 in quantities of 100 or more. The design of the paged memory may change to accommodate new advances in CCD technology. For example, Texas Instruments recently introduced a 64K-bit CAM memory.
- The work memory has been designed using the 64 bit (8 by 8) CAM SCL5533 manufactured by Solid State Scientific, Inc. This integrated circuit costs about $20 in units of 100 or more.
- The page bits of the work memory have been designed using components similar to the Fairchild 93410 integrated circuit which is a 256-bit (256 by 1) RAM. This integrated circuit costs about $16.80.

The microprocessor controller has not been completely specified. Some basic requirements include a capability for generating control signals for the system, a capability for communicating with the host computer (initially a PDP 11), and an execution speed sufficient for decoding host requests into commands fast enough to warrant implementing this proposed associative memory system. The microprocessor must have a program interrupt capability to

most efficiently monitor the operation of the associative memory. The microprocessor must be capable of communicating with selected output devices while continuing to be receptive to data from the host and from the associative memory. The microprocessor must have a word length of at least eight bits to handle the 8-bit addresses required for the associative memory. The estimated cost for the microprocessor is $700.

The logic integrated circuits are used to implement the response resolver, the CAM-to-CCD-to-microprocessor interfaces, and the page control. These integrated circuits use transistor-transistor logic (TTL) that represents small scale integration (SSI) and medium scale integration (MSI) fabrication techniques. These components consist of logic gates (NAND, NOR, AND, OR), flip flops, storage registers, shift registers, counters, digital multiplexers and decoders, and many others. About a thousand integrated circuit components are required, costing about $3 each.

Pin Cost

The pin cost of the system is the summation of the number of pins on the integrated circuits that are required to implement the system. The pin cost estimate also covers other fabrication costs such as power supplies, printed circuit boards, and cooling and testing costs. The cost assessment per pin is 30¢.[3]

Cable Interconnection Cost

Cables are required for transmitting data and for providing control signals. The cabling cost is dependent on the number of connectors involved. The system requires an estimated 100 connectors costing about $1.00 each. The cabling cost is assessed at $4 per connector.

[3]White, L.S. and T.A. Welch, *"Analysis of Virtual Memory Implementations,"* Technical Report 174, Electronics Research Center, The University of Texas, Austin, Texas (Jul 1975).

CONCLUSIONS

The hardware paged associative memory system offers an attractive method for implementing an associative (content-addressable) memory capability. The system proposed is both flexible and cost effective. The system represents an improvement over software simulations because it allows interrogations to be performed on any of the bits of the data field. This improvement allows data field lengths to vary depending on the application and on the bit precision required by the application. This capability also allows directed graphs to be parsed in many ways. For example, interrogations may produce links, nodes, or node-link combinations.

The mass memory portion of the associative memory can be implemented using any one of several memory technologies available—a CCD, a magnetic bubble, or disk memories—or some combination of these.

Each of the different memory technologies has its advantages and disadvantages. Disk memories are inexpensive and nonvolatile, but they are slow due to large amounts of mechanical access. Magnetic bubble memories are also nonvolatile, and offer a shorter access time than disk, but they are not currently available as a production item. CCD memories have a faster access time than bubble memories and are readily available as a production item, but they are volatile, and require that a source of standby electrical power be available so that a loss of power will not require the memory to be reloaded when the electrical power comes back on.

Packaging considerations figure importantly in the implementation of a classifier. Both CCD and magnetic bubble memories have a packaging advantage over disk memories. Disk memories require hardware for mechanical rotation and positioning. Both CCD and magnetic bubble memories can be mounted on printed circuit boards that would fit in the associative memory chassis. Magnetic bubble memories promise to have two to four times the packing density per chip, and thus would require fewer chips for a given storage capacity. However the cost per bit for both CCD and magnetic bubble memories is projected to be about the same.

The mass memory for the proposed associative memory would be implemented using CCD technology. The CCD technology has been selected because it has fast access times and transfer rates, and it is commercially available and economical.

The options as to the amount of main memory necessary have been shown in Figure 17 and Table 6. The most cost effective alternatives use 256- to 1024-cell CAM for main memory. These options offer acceptable retrieval times with costs ranging from $25.5K to $55K.

The 256-cell CAM has been chosen for the design because it offers an acceptable retrieval time at a cost which is reasonable for the construction of a prototype device.

The hardware system proposed is a cost effective alternative to the software simulation. The performance of the software simulation on the PDP 11 computer is indicated in Figure 17 by the points marked "C" and "S". The point marked "C" represents the simulation with core memory, and "S" represents projected simulation with semiconductor memory. The average retrieval time using the core memory simulation is 570 microseconds. The cost for core memory storage equivalent to that offered by the 65K-cell associative memory is about $20K. Thus the proposed hardware paged system will operate about 15 times as fast as the core memory simulation and will cost only 1.25 times more. The hardware system has a price performance advantage of 12:1 over the core simulation.

If the simulated system used semiconductor memory, the estimated retrieval time would be 170 microseconds. The cost of the semiconductor memory would be about $35K. The hardware paged system has a price performance advantage of 6.3:1 over a semiconductor memory simulation because the hardware system operates 4.6 times as fast.

The proposed design is tentative and further refinements will influence the final design. These refinements include decreasing bubble memory costs, refining the multiple response resolver design, and analyzing further the acoustic parsing algorithms.

36 Blank

INITIAL DISTRIBUTION

Copies		Copies	Code	
2	Defense Communications Engineering Center 1 R720 G. Gluck 1 R740	1	185	T. Corin
		1	189	G. Gray
		30	5214.1	Reports Distribution
1	CHONR/430D M. Denicoff	1	522.1	Lib (C)
1	NRL	1	522.2	Lib (A)
1	NAVSUP/0414 G. Bernstein			
1	NSWC			
1	NUSC			
1	NOSC			
3	NAVSEC 1 SEC 6114 1 SEC 6178D			
1	Rome Air Development Center			
12	DDC			

CENTER DISTRIBUTION

Copies	Code	
1	0120	R. Allen
1	18/1808	G. Gleissner
1	1802.2	F. Frenkiel
1	1802.4	F. Theilheimer
1	1805	E. Cuthill
1	1809	D. Harris
1	182	A. Camara
30	1824	J. Carlberg
1	184	H. Lugt

DTNSRDC ISSUES THREE TYPES OF REPORTS

(1) DTNSRDC REPORTS, A FORMAL SERIES PUBLISHING INFORMATION OF PERMANENT TECHNICAL VALUE, DESIGNATED BY A SERIAL REPORT NUMBER.

(2) DEPARTMENTAL REPORTS, A SEMIFORMAL SERIES, RECORDING INFORMATION OF A PRELIMINARY OR TEMPORARY NATURE, OR OF LIMITED INTEREST OR SIGNIFICANCE, CARRYING A DEPARTMENTAL ALPHANUMERIC IDENTIFICATION.

(3) TECHNICAL MEMORANDA, AN INFORMAL SERIES, USUALLY INTERNAL WORKING PAPERS OR DIRECT REPORTS TO SPONSORS, NUMBERED AS TM SERIES REPORTS; NOT FOR GENERAL DISTRIBUTION.